(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号
特開2000－298652
（P2000－298652A）
(43)公開日　平成12年10月24日(2000.10.24)

| (51)Int.Cl.' | | 識別記号 | FI | | テーマコード'(参考) |
|---|---|---|---|---|---|
| G06F | 15/16 | 640 | G06F 15/16 | 640B | 5B045 |
| | 15/78 | 510 | 15/78 | 510G | 5B062 |
| | | | | 510A | |

審査請求　未請求　請求項の数7　ＯＬ　（全 8 頁）

(21)出願番号　特願平11－106052

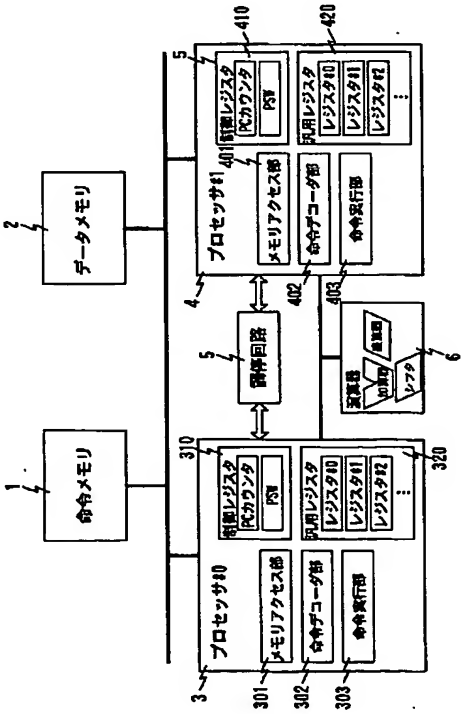
(22)出願日　平成11年４月14日(1999.4.14)

(71)出願人　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目２番３号
(72)発明者　鈴木　和雅
東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
(72)発明者　服部　孝
東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
(74)代理人　100099461
弁理士　溝井　章司　（外２名）

Ｆターム(参考)　5B045 EE01 EE03 EE14 KK08
5B062 CC04 DD02 DD05 DD10

(54)【発明の名称】　マルチプロセッサ

(57)【要約】
【課題】　単体のプロセッサよりも複数のプロセスを効率良く実行できるようにし、従来のマルチプロセッサよりも面積を小さくする。
【解決手段】　複数のプロセッサ３、４と、複数のプロセッサから共有される命令メモリ１及びデータメモリ２と、複数のプロセッサから共有される演算器６を備え、更に命令メモリ１、データメモリ２、及び演算器６が同時期には１つのプロセッサからのみアクセスされるように調停する調停回路５を備える。



1

【特許請求の範囲】

【請求項１】　複数のプロセッサと、前記複数のプロセッサから使用される共有演算器と、前記複数のプロセッサのうち一のプロセッサが前記共有演算器を使用している間は他のプロセッサが前記共有演算器を使用できないように制御する調停回路を備えることを特徴とするマルチプロセッサ。

【請求項２】　前記調停回路は、前記一のプロセッサの前記共有演算器の使用を許可する信号を前記一のプロセッサに送出するとともに、前記他のプロセッサの前記共有演算器の使用を拒絶する信号を前記他のプロセッサに送出することを特徴とする請求項１記載のマルチプロセッサ。

【請求項３】　前記調停回路は、前記共有演算器を使用できる前記一のプロセッサと前記他のプロセッサとを一定時間ごとに切り替えることを特徴とする請求項１記載のマルチプロセッサ。

【請求項４】　複数のプロセッサと、前記複数のプロセッサから使用される共有演算器と、前記複数のプロセッサから使用される共有メモリと、前記複数のプロセッサのうち一のプロセッサが前記共有演算器を使用している間は他のプロセッサが前記共有演算器を使用できないように制御するとともに前記複数のプロセッサのうち一のプロセッサが前記共有メモリを使用している間は他のプロセッサが前記共有メモリを使用できないように制御する調停回路を備えることを特徴とするマルチプロセッサ。

【請求項５】　前記調停回路は、前記共有演算器の使用については前記一のプロセッサの前記共有演算器の使用を許可する信号を前記一のプロセッサに送出するとともに、前記他のプロセッサの前記共有演算器の使用を拒絶する信号を前記他のプロセッサに送出し、前記共有メモリの使用については前記一のプロセッサの前記共有メモリの使用を許可する信号を前記一のプロセッサに送出するとともに、前記他のプロセッサの前記共有メモリの使用を拒絶する信号を前記他のプロセッサに送出することを特徴とする請求項４記載のマルチプロセッサ。

【請求項６】　前記調停回路は、前記共有演算器の使用については前記共有演算器を使用できる前記一のプロセッサと前記他のプロセッサとを一定時間ごとに切り替え、前記共有メモリの使用については前記共有メモリを使用できる前記一のプロセッサと前記他のプロセッサとを一定時間ごとに切り替えることにより制御することを特徴とする請求項４記載のマルチプロセッサ。

【請求項７】　前記共有演算器と前記複数のプロセッサとが前記共有メモリと前記複数のプロセッサとの間の接続と別個に接続されていることを特徴とする請求項４記載のマルチプロセッサ。

【発明の詳細な説明】

【０００１】

2

【発明の属する技術分野】この発明は、複数のプロセッサを実装したマルチプロセッサに関するものである。特に、単一のＬＳＩ上に複数のプロセッサを実装した１チップマルチプロセッサにおいて、複数のプロセッサで回路を共有することによりＬＳＩ面積を抑えつつ、複数のプロセスを効率よく実行することが可能なマルチプロセッサに関するものである。

【０００２】

【従来の技術】単一のプロセッサからなるシングルプロセッサ上で複数の処理を実行する手段として代表的なものに、ＯＳ（基本ソフトウェア）をベースにおき、ＯＳ上で各種アプリケーションを実行するというものがある。ＯＳはあるプログラムの処理単位（以後、プロセスと呼ぶ）に分けて各アプリケーションを実行する。同時に１つのプロセッサ上で実行できるプロセスは１つであるため、ある時間毎に実行するプロセスを細かく切り替えることによって、あたかも複数のプロセスを同時に実行しているかのように見せている。

【０００３】図９は、２つのプロセスが時分割で実行されているときの、プロセッサ上で動作している処理の様子を示したもので、４００１、４００２はそれぞれ異なる２つのプロセス　プロセスＡ、プロセスＢを示し、４００３は、２つのプロセスの切り替え（以後、「プロセススイッチ」という）のために必要な処理を示している。プロセスを時分割で切り替える一般的な手段としては、タイマーからの割り込みを利用する方法がある。例えば、タイマーからの割り込み回数をカウントしておき、あらかじめ決められたあるカウント値に達したら、プロセススイッチを行うようにする。プロセススイッチでは、プロセッサの視点から見た場合には、切り替える前に実行していたプロセスに関するデータ（プログラムカウンタ（ＰＣ）値、スタックポインタアドレス、汎用レジスタ、プログラム状態語（ＰＳＷ）中の必要な情報など）のメモリへの待避、切り替え先のプロセスに関するデータ（ＰＣ値、スタックポインタ、汎用レジスタ、ＰＳＷ内の情報など）のデータメモリからのリストアが行われる。そのため、プロセススイッチが実行されると、プロセッサでは数十～数百の命令が実行されることになり、次のプロセスへの切り替えに時間を要し、高速処理が図れないという問題があった。

【０００４】この問題を解決し、種々の処理を高速に実行する手法の１つとして、マルチプロセッサシステムがある。これは、必要とされる処理を、複数のプロセッサに分散して並列に処理することによって、高速化を計るものである。近年のＬＳＩ集積技術は、このマルチプロセッサシステムを単一のＬＳＩ上に実装することを可能にし、マルチプロセッサシステムを安価に実現することを可能にした。しかし、マルチプロセッサは、高速処理が図れる一方で、複数のプロセッサが各々命令メモリ等のハードウェア・リソースを有するため面積的に不利と

いう問題点がある。この問題を解決する手段として、例えば、特開昭６２－２４３５６で開示された１チップマイクロコンピュータがある。特開昭６２－２４３５６では、特にＲＯＭ、ＲＡＭを共用することにより面積的に有利にするとともに、各プロセッサでのＲＯＭ、ＲＡＭをアクセスするタイミングをずらすことにより効率的にプログラムを実行することが可能なマルチプロセッサシステムについて開示している。

【０００５】

【発明が解決しようとする課題】以上のように、従来の単一のプロセッサで複数のプロセスを実行する際には、プロセススイッチが発生してしまう。このため、ユーザからは実際には必要としない処理に対して、時間を消費してしまうことになり、高速処理が実現できないという問題がある。マルチプロセッサでは、各プロセッサで実行するプロセスを分けることにより、上記の問題を解決することが可能であるが、それぞれのプロセッサが独立してハードウェア・リソースを持つため、面積的に不利との問題点があった。この点につき、従来はＲＯＭ、ＲＡＭといったメモリを共有する手段を採用していたが、近年のプロセッサの省面積要求に対して、このメモリの共有のみでは十分とはいえず、更なる省面積化を実現するマルチプロセッサ技術が望まれていた。そこで、この発明は上記のような問題点を解決するためになされたもので、複数のプロセスが実行されるときにも効率よく実行することができ、かつ省面積なプロセッサを実現することを目的とする。

【０００６】

【課題を解決するための手段】この発明に係るマルチプロセッサは、複数のプロセッサと複数のプロセッサから共有されるメモリ、更には複数のプロセッサから共有される演算器を備えるものであり、メモリ、及び演算器は同時期には一つのプロセッサからのみアクセスされるように調停する手段を備えるものである。具体的には、以下の特徴を有する。

【０００７】命令コードを記憶する命令メモリと、被演算データや演算結果を記憶するデータメモリと、演算器を内部に含まない複数のプロセッサと、プロセッサからの制御データに従って演算を行う演算器とを持ち、前記命令メモリ、前記データメモリ、及び前記演算器は、複数の前記プロセッサによって共有され、それぞれ同時期には多くとも一つの前記プロセッサのみからアクセスされるようにする手段を持つことを特徴とする。

【０００８】一定の期間ごとに、命令メモリ、データメモリ、及び、演算器を使用することが可能なプロセッサを切り替える手段を持つことを特徴とする。

【０００９】

【発明の実施の形態】実施の形態１．図１、図２はこの発明の実施の一形態を示す１チップマルチプロセッサの構成図である。説明を簡単にするため、ここではプロセッサが２つの場合を例にとって説明する。図において、１は命令コードを記憶する命令メモリ、２は、被演算データや命令の実行結果などを記憶するデータメモリ、３は命令コードに従って命令を実行するプロセッサ＃０であり、内部に演算器を含んでいない。４はプロセッサ＃１であり、プロセッサ＃０（３）と同様に内部に演算器を含んでいない。５は調停回路であり、命令メモリ１、データメモリ２及び演算器６をどちらのプロセッサが使用するかを決定する。つまり、調停回路５は、共有メモリ（命令メモリ及びデータメモリ）のみならず、演算器６の使用についても調停する。６は演算器であり、演算命令を実行する際に必要となる加算器や乗算器などを含む。また、演算器はプロセッサ＃０（３）及びプロセッサ＃１（４）に共有されている。３０１、４０１はメモリアクセス部であり、命令コードを命令メモリ１から読み出す（フェッチする）ときや、ロード／ストア命令において、レジスタとメモリ間でデータ転送を行うときに、命令メモリ１やデータメモリ２に対してアクセスを行う。３０２、４０２は命令デコーダ部であり、読み出した（フェッチした）命令コードをデコードし、適切な制御信号を命令実行部３０３、４０３や調停回路５に伝える。３０３、４０３は命令実行部であり、命令デコーダ部３０２や調停回路から送出された制御信号に従って、演算器６に制御信号や必要なデータを送出したり、演算器６が実行した演算結果をラッチして汎用レジスタ３２０、４２０に書込んだりする。また、３１０、４１０は制御レジスタであり、現在実行中のプログラムの次に読み出す（フェッチする）命令コードの命令メモリ１上のアドレスを保持するプログラムカウンタや、プロセッサの状態などの情報を保持するプログラムステータスワード等から構成される。３２０、４２０は汎用レジスタであり、演算結果データや被演算データなどを一時的に保持するレジスタから構成される。図１では、命令メモリ１、データメモリ２、及び演算器６が共通のバスを介してプロセッサ＃０（３）とプロセッサ＃１（４）に接続している構成を示してしている。図２においては、演算器６は、命令メモリ１、データメモリ２とは別のバスを介して各プロセッサに接続されている。図１の構成ではバスを共通にするため、バスの数を減少させることができるとのメリットがある。一方で、図１では、バスが共通するためデータの送受信が混み合うとのデメリットがあり、図２の構成は演算器６と共有メモリ（命令メモリ及びデータメモリ）で使用するバスを別にしているのでデータ送受信が円滑に行えるとのメリットがある。これらの構成は、プロセッサの使用目的や使用条件等によって選択することができる。

【００１０】図３は、プロセッサ＃０（３）及びプロセッサ＃１（４）における演算命令のパイプラインの構成を示したもので、４段のパイプラインから構成されている。図３において、２０は命令を読み出す（フェッチす

る）期間（以下、「ＩＦステージ」という）であり、２１は、命令をデコードする期間（以下「Ｄステージ」という）であり、２２は、デコードした命令を実行する期間（以下「Ｅステージ」という）であり、２３は、演算した結果をレジスタに書き戻す期間（以下、「Ｗステージ」という）である。次に動作について説明するが、プロセッサ＃０（３）もプロセッサ＃１（４）も基本的な動作は同じなので、特にことわりがない限り、プロセッサ＃０（３）に関して説明していく。ＩＦステージ２０では、ＰＣカウンタが示すアドレスにある命令コードをメモリアクセス部３０１を経由して命令メモリ１から読み出す。次に、Ｄステージ２１では、読み出した命令コードを命令デコーダ部３０２にてデコードし、デコード結果を命令実行部３０３に送出する。このとき、実行する命令コードが演算命令の場合には、調停回路５に対して、次のステージで演算器６へのアクセスを行う旨のリクエスト信号を送出する。そして、Ｅステージ２２では、命令デコーダ部３０２から出力された制御信号、及び、調停回路５からのアクセスを許可することを意味するアクノーリッジ信号に従って、演算器６に対して被演算データ及び演算の種別を示す制御コードを送出する。演算器６では、受け取ったデータ及び制御コードに従って演算を行った後、命令実行部３０３に演算結果を送出する。最後に、Ｗステージ２３では、命令実行部３０３は、演算器６から受け取った演算結果を、汎用レジスタ３２０に書込む。調停回路５では、プロセッサ＃０（３）、プロセッサ＃１（４）が命令メモリ１、データメモリ２、演算器６の各リソースを使用するときの調停を行う。リソースを使用する側のプロセッサは、調停回路５に対して、使用する各リソースに応じたリクエスト信号を送出し、それを受け取った調停回路５は、使用の許可を意味するアクノーリッジ信号を、使用を許可するプロセッサに対して送出する。調停回路５では、各リソースに対して、先にリクエスト信号を出したプロセッサに対して、アクノーリッジ信号を送出するが、複数のプロセッサから同時期にリクエスト信号が送出されたときには、優先度に従ってアクノーリッジ信号を送出し、他方のプロセッサに対してはアクセスを拒絶する信号を送出する。この優先度の付け方に関しては、例えば、固定にしてもよいし、順番に各プロセッサ間で優先度が変るようにしてもよい。

【００１１】図４は、プロセッサ＃０（３）、プロセッサ＃１（４）で、それぞれ異なる一連の演算命令を実行したときのパイプラインの様子を示したものである。３０、３２、３４は、プロセッサ＃０（３）において、この順番に実行される一連の演算命令であり、３１、３３、３５は、プロセッサ＃１（４）において、この順番に実行される一連の演算命令である。プロセッサ＃０（３）、プロセッサ＃１（４）から同時期に命令メモリ１のリクエスト信号を調停回路５に対して出力し、調停



回路５はプロセッサ＃０（３）に対してアクノーリッジ信号を返したため、プロセッサ＃１（４）側で実行される演算命令３１のＩＦステージは、プロセッサ＃０（３）側の演算命令３０のＩＦステージが完了するまで待たされ、２クロックサイクル分続いている。同様に、プロセッサ＃０（３）で実行される演算命令３２も、プロセッサ＃１（４）で実行される演算命令３１のＩＦステージが完了するまで待たされるために２クロックサイクル分続き、以下、演算命令３３、３４、３５に関しても同様である。次に、演算命令３２のＥステージにおいては、プロセッサ＃０（３）は調停器５からアクノーリッジ信号を受けて演算器６を使用しているが、演算処理に２クロックサイクル要している。このため、演算命令３２のＥステージと演算命令３３のＥステージとが重複することになる。この場合プロセッサ＃１（４）は、演算器６へアクセスするためのリクエスト信号を調停回路５に対し送出するが、演算器６は既にプロセッサ＃１（０）により使用されているためプロセッサ＃１（４）のリクエストは拒絶され、演算命令３２のＥステージが完了するまで待たされる。このため、プロセッサ＃１（４）の演算命令３３のＥステージは２クロックサイクル分続き、以下、演算命令３４，３５に関しても同様に２クロックサイクル続くことになる。

【００１２】図５は、本実施の形態における１チップマルチプロセッサにおいて、２つのプロセスを実行したときの様子を示したものである。４１は、プロセッサ＃０（３）で、４２は、プロセッサ＃１（４）でそれぞれ実行されているプロセスを示す。また、４０はシングルプロセッサにおいて時分割制御により同様の処理を実行したときの様子を示しており、４１、４２との比較のために示している。この実施の形態における１チップマルチプロセッサでは、複数のプロセッサが使用するリソースの競合が発生した場合に、対応するパイプラインステージが延びるため、プロセスＡ（４００１）、プロセスＢ（４００２）とも単体では処理時間が長くなっているが、競合しないとき、あるいは、命令メモリ、データメモリ、演算器を使用しないステージに関しては２つのプロセッサで並列に処理を実行することが可能であるため、また、プロセススイッチ４００３が発生しないため、総合的な処理時間は大幅に短縮されている。また、本発明の調停回路はメモリのみならず、演算器６についての調停も行うことができるため、各プロセッサで演算器６を共有することができる。この結果、各プロセッサが演算器を内部に搭載するときに比べて大幅に面積を減少させることができる。特に、多数のプロセッサが配置される場合には省スペース化に大きく貢献することになる。

【００１３】以上のように、複数のプロセスを実行したときにプロセススイッチが発生せずに効率よく処理を実行することができ、またメモリや演算器を複数のプロセ

ッサで共用するように構成したので、省スペースな１チップマルチプロセッサを実現することができる。

【００１４】実施の形態２．以上の実施の形態１では、複数のプロセッサに共用されているリソース（命令メモリ１、データメモリ２、演算器６）を各プロセッサが使用する際に、調停回路に対してリクエストを出し、調停回路から使用を許可されたプロセッサが、目的のリソースを使用するようにしたものであるが、次に一定の時間間隔でリソースを使用することができるプロセッサを切り替えるようにする場合の実施の形態を示す。この実施の形態では、調停回路５はタイマーカウンタを持ち、ある一定時間間隔毎にリソースを使用することを許可するプロセッサを切り替える。つまり、命令メモリ１、データメモリ２のみならず、演算器６についても、使用するプロセッサを時間ごとに切り替えるものである。図６は、この実施の形態におけるプロセッサと調停回路との信号の送受信の動作を説明するためのブロック図で、７、８は、調停回路５からそれぞれプロセッサ＃０（３）、プロセッサ＃１（４）に出力される制御信号であり、９、１０は、それぞれプロセッサ＃０（３）、プロセッサ＃１（４）から調停回路５に出力される制御信号である。制御信号７、８は、それぞれのプロセッサに対して、現在リソースを使用することを許可するか否かを示す信号であり、制御信号９、１０は、それぞれのプロセッサが、現在、リソースを使用しているか否かを示す信号である。

【００１５】次に動作について説明する。タイマーカウンタがある一定の値に達するとタイマーカウンタはリセットされ再びカウントを開始するとともに、調停回路５は、プロセッサ＃０（３）に対して制御信号７をアサートする。プロセッサ＃０（３）は、制御信号７がアサートされると、調停回路５に対して、制御信号９をアサートする。このとき、制御信号８、１０はネゲートされたままである。さらにタイマーカウンタが再びある一定の値に達するとタイマーカウンタは再びリセットされ、調停回路５は、プロセッサ＃０（３）に対する制御信号７をネゲートする。プロセッサ＃０（３）は、制御信号７がネゲートされると、現在、処理中の命令を一時停止して、リソースの使用をやめるとともに、制御信号９をネゲートする。調停回路５は、制御信号９がネゲートされると、制御信号８をアサートし、プロセッサ＃１（４）に対してリソースを使用することを許可する。プロセッサ＃１（４）は、制御信号８がアサートされると、制御信号１０をアサートし再び制御信号８がネゲートされるまで、処理を実行する。以下、同様にして、プロセッサ＃０とプロセッサ＃１が交互にある時間間隔毎に処理を実行していく。図７は、本実施の形態におけるパイプラインを示している。ＩＦ、Ｄ、Ｅ、Ｗの各ステージ及び３０、３１、３２、３３、３４、３５の各演算命令は実施の形態１．の場合と同様である。一定時間ごとにプロセッサ＃０（３）とプロセッサ＃１（４）が交互に共有リソース（命令メモリ１、データメモリ２、演算器６）を使用し、一方のプロセッサが共有リソースを使用している間は、他方のプロセッサは共有リソースを使用していない。図８は、この実施の形態における１チップマルチプロセッサにおいて、２つのプロセスを実行しているときの様子を示したものである。４３は、プロセッサ＃０（３）で、４４は、プロセッサ＃１（４）でそれぞれ実行されているプロセスを示す。また、４０はシングルプロセッサにおいて時分割制御により同様の処理を実行したときの様子を示しており、４１、４２との比較のために示している。４００４は、一方のプロセッサが動作している間に他方が処理を一時停止している期間である。この実施の形態の場合、２つのプロセス自体を処理するのに要する時間は、単一のプロセッサで実行した場合と同一であるが、プロセススイッチが発生しないため、総合的な処理時間は短くなっている。以上のように、一定時間毎に、動作するプロセッサを切り替えるように構成したため、プロセススイッチが発生せずに、効率よく処理を実行することが可能である。また、各プロセッサからのリソース使用の要求の競合が発生しないため、調停回路、及び、プロセッサのタイミング制御を単純な回路で実現することが可能である。また、一時停止しているプロセッサに対してはクロックの供給を止めるなどして、消費電力を低減させることが可能である。

【００１６】

【発明の効果】以上のように、この発明の１チップマルチプロセッサによれば、複数のプロセスを実行したときにプロセススイッチが発生せずに効率よく処理を実行することができ、またメモリのみならず演算器をも複数のプロセッサで共用するように構成したので、省スペースな１チップマルチプロセッサを実現することが可能である。また、メモリのみならず演算器についても、タイマーにより一定時間間隔で、使用するプロセッサを完全に切り替えるので、効率よく処理を実行することができる省スペースな１チップマルチプロセッサを、容易に実現することが可能である。

【図面の簡単な説明】

【図１】　この発明の１チップマルチプロセッサを示すブロック図。

【図２】　この発明の１チップマルチプロセッサを示すブロック図。

【図３】　この発明の１チップマルチプロセッサにおける演算命令のパイプライン図。

【図４】　この発明の１チップマルチプロセッサにおいて、２つのプロセッサで同時に演算命令を実行したときのパイプライン図。

【図５】　この発明の実施の形態１における１チップマルチプロセッサにおいて、２つのプロセスを実行したときの動作図。

9

【図６】　この発明の実施の形態２における１チップマルチプロセッサにおける調停回路およびプロセッサの構成図。

【図７】　この発明の１チップマルチプロセッサにおいて、２つのプロセッサで同時に演算命令を実行したときのパイプライン図。

【図８】　この発明の実施の形態２における１チップマルチプロセッサにおいて、２つのプロセスを実行したときの動作図。

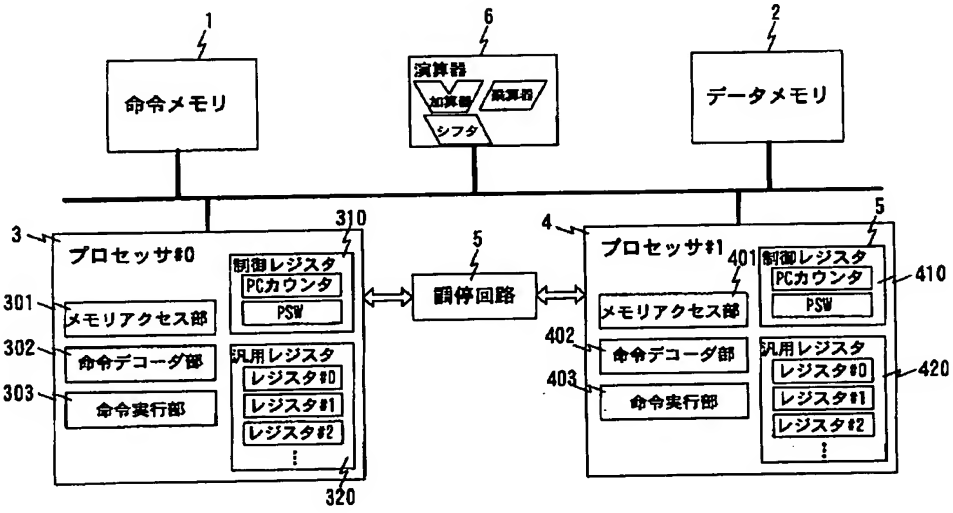
【図９】　従来の単一のプロセッサにおいて、２つのプロセスを実行したときの動作図。

【符号の説明】

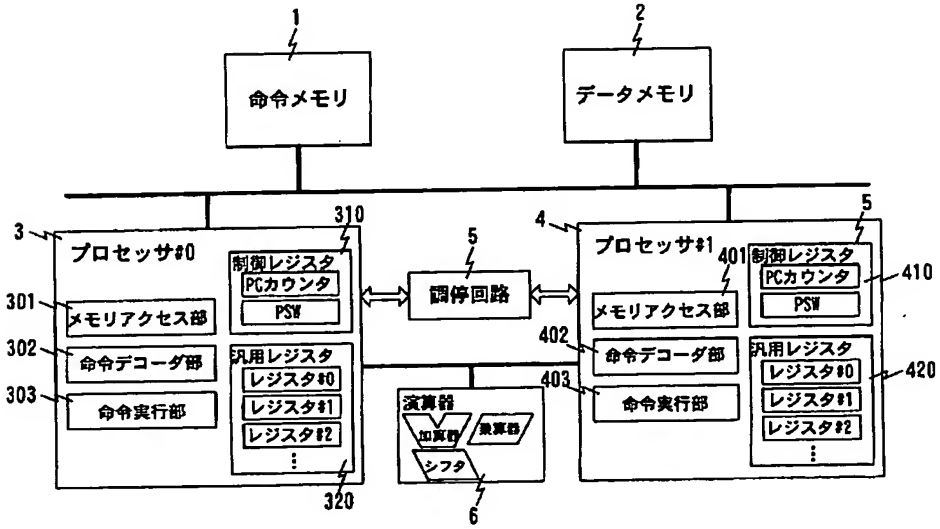
10

１　命令メモリ、２　データメモリ、３　プロセッサ＃０、３０１　メモリアクセス部、３０２　命令デコーダ部、３０３　命令実行部、４　プロセッサ＃１、４０１　メモリアクセス部、４０２　命令デコーダ部、４０３　命令実行部、５　調停回路、６　演算器、２０　パイプラインのＩＦステージ、２１　パイプラインのＤステージ、２２　パイプラインのＥステージ、２３　パイプラインのＷステージ、３０～３５　演算命令、４０～４４　プロセス処理、４００１　プロセスＡ、４００２　プロセスＢ、４００３　プロセススイッチ、４００４　動作一時停止期間。

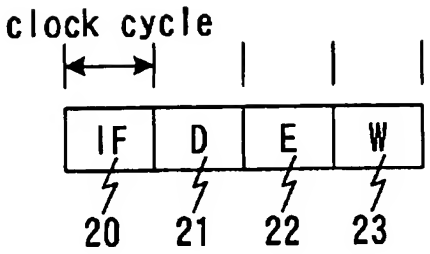
【図１】



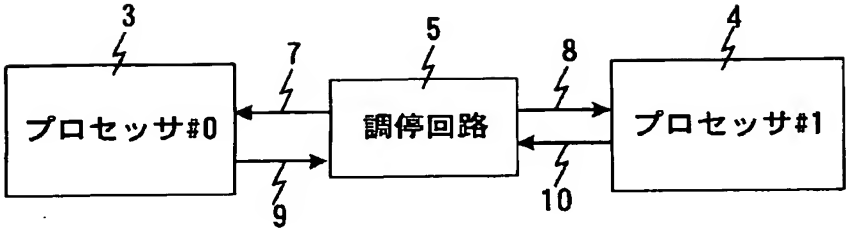

【図２】



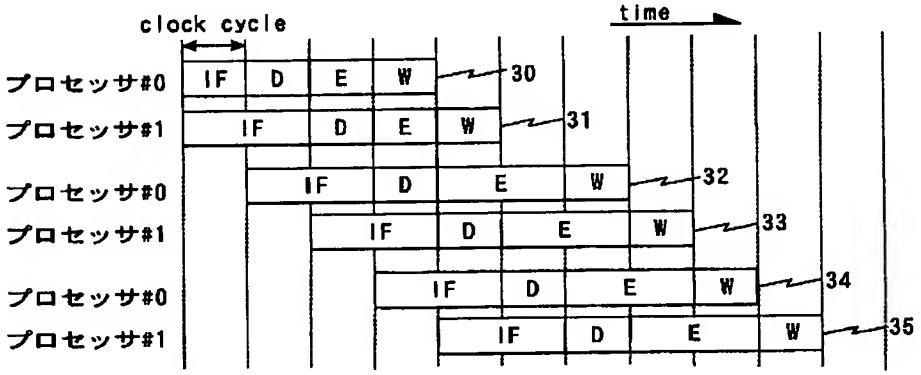
【図３】


clock cycle
IF
D
E
W
20
21
22
23

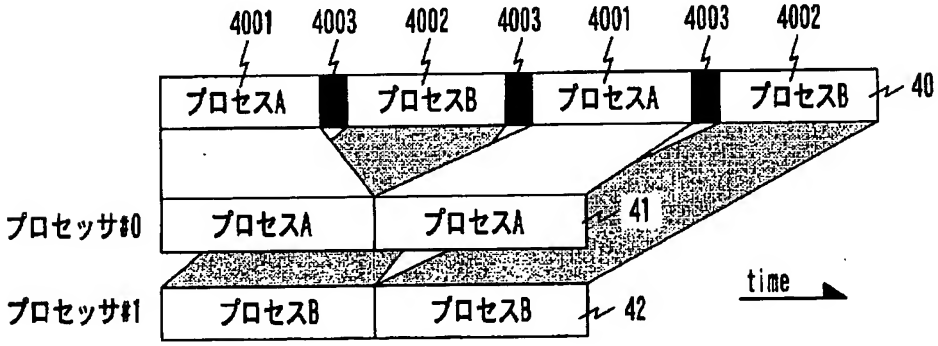
【図６】


3
プロセッサ#0
7
9
5
調停回路
8
10
4
プロセッサ#1

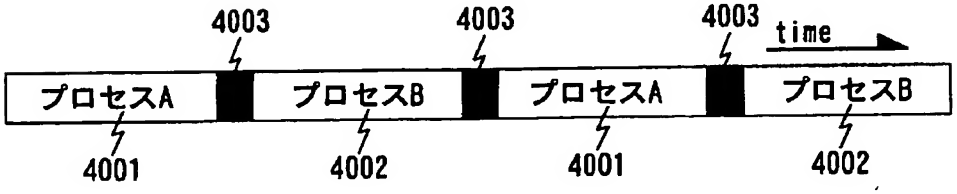
【図４】


clock cycle
time
プロセッサ#0
プロセッサ#1
プロセッサ#0
プロセッサ#1
プロセッサ#0
プロセッサ#1
IF
D
E
W
30
31
32
33
34
35

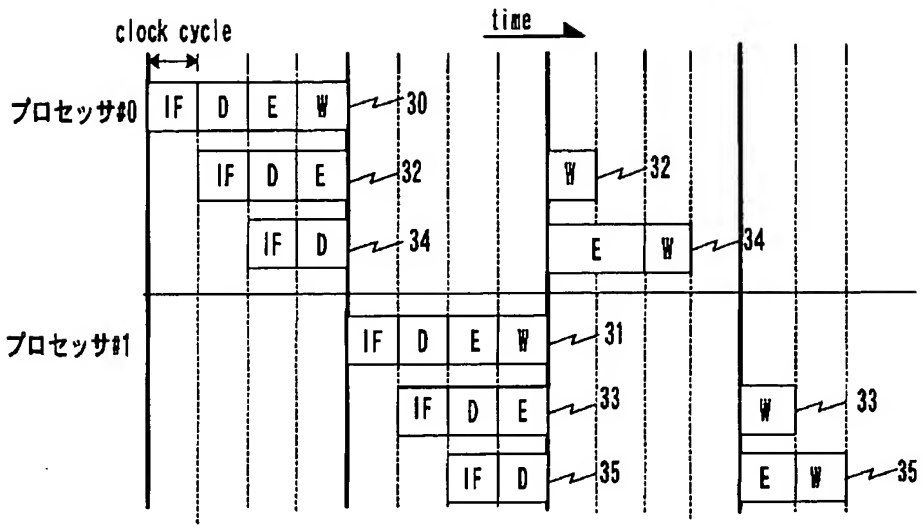
【図５】


4001
4003
4002
4003
4001
4003
4002
プロセスA
プロセスB
プロセスA
プロセスB
40
プロセッサ#0
プロセスA
プロセスA
41
time
プロセッサ#1
プロセスB
プロセスB
42

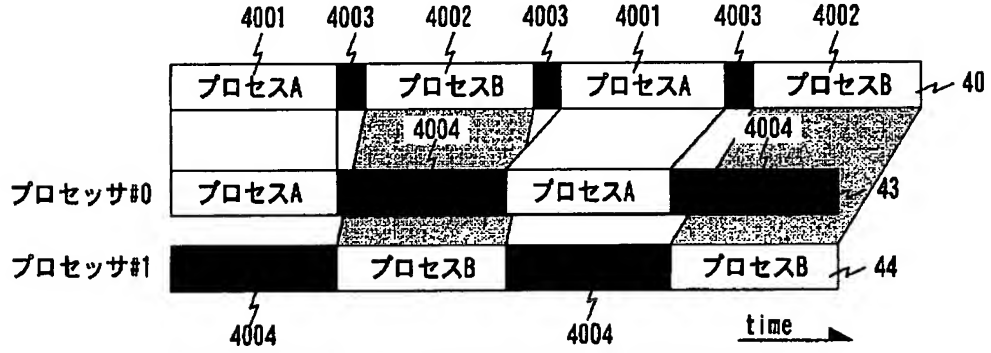
【図９】


4003
4003
4003
time
プロセスA
プロセスB
プロセスA
プロセスB
4001
4002
4001
4002

【図７】


clock cycle
time
プロセッサ#0
プロセッサ#1
IF
D
E
W
30
32
34
31
33
35

【図８】


4001
4003
4002
4003
4001
4003
4002
プロセスA
プロセスB
プロセスA
プロセスB
40
4004
4004
プロセッサ#0
プロセスA
プロセスA
43
プロセッサ#1
プロセスB
プロセスB
44
4004
4004
time